Resource Allocation and Population Size in MOEA/D

Yuri Lavinas[†], Claus Aranha[†], Marcelo Ladeira[††], Tetsuya Sakurai[†]
筑波大学[†], ブラジリア大学[††]

1 Introduction

Multi-objective Optimization Problems (MOPs) appear in problems with two or more conflicting objective functions that need to be simultaneously optimized. Finding good sets of solutions for MOPs is considered a hard issue problem for which Evolutionary Algorithms have been proposed as potential solvers [1–3].

The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [4] is a practical algorithm for solving MOPs. The key idea of MOEA/D is to decompose the multi-objective optimization problem into a set of single-objective subproblems, which are solved simultaneously.

While the original MOEA/D did not discriminate between subproblems, it has since become clear that focusing computational effort on specific subsets of these subproblems can substantially improve the performance of the algorithm [5–9].

Several works have proposed to address this issue and to investigate methods to allocate different amounts of computational effort to subproblems [5, 6, 8–12]. These approaches, known as Resource Allocation (RA) techniques, have been shown to result in consistent performance improvements for the MOEA/D. While different RA techniques have their particular characteristics, all limit the number of solutions that are updated at a given iteration.

This raises the question: are the improvements from RA in practice due to a reduction of the population size? A recent paper by Pruvost et al. [9] points that way, where they explore different population sizes in MOEA/D in discrete problems. However, we feel that other factors are also at work in this question.

With that in mind, in this work we investigate and quantify the effects of RA on MOEA/D against MOEA/D with different population sizes on continuous problems. To achieve this, we study the correspondences of MOEA/D using RA against: (1) no RA on a small population size and (2) no RA on a larger population size.

From an algorithm mechanics point of view, MOEA/D with RA is comparable to MOEA/D with a small population size, because both select to update few solutions every iteration. It is also comparable to MOEA/D with big population size, as both maintain a large population set. On the other hand, MOEA/D with RA is different from MOEA/D with a small or large population size because it only updates a (different) subset of its population at every iteration, while the other two variants update their whole population.

The result of these similarities and differences is that MOEA/D with RA will show a hybrid behavior on our experiments: in general, it converges to the Pareto front at about the same speed as MOEA/D with small population, but it explores the search space more like an MOEA/D with a large population. This motivates the use of RA as an approach to mitigate common problems related to the choice of population size.

The remainder of this paper is organized as follows: Section 2 reviews the main concepts related to resource allocation in the MOEA/D. Section 3 explains the relationship between the choice of population size and resource allocation in MOEA/D. Section 4 describes the MOEA/D with n-partial update strategy. Section 5 presents experimental results related to the investigation of the effect of partial updates on the performance of the algorithm, as well as comparisons between

† Yuri Lavinas(lavinas.yuri.xp@alumni.tsukuba.ac.jp)
† Claus Aranha(caranha@cs.tsukuba.ac.jp)
†† Marcelo Ladeira(mladeira@unb.br)
† Tetsuya Sakurai(sakurai@cs.tsukuba.ac.jp)
Department of Computer Science, University of Tsukuba (†)
Department of Computer Science, University of Brasilia (††)

MOEA/D-PS against two different configurations of MOEA/D, one with a small population size and another with a big population size. Finally, Section 7 presents our concluding remarks.

2 Resource Allocation

The key idea behind MOEA/D is to decompose a MOP into a set of single-objective subproblems, which are then solved simultaneously. While these subproblems are usually considered equivalent, a growing body of work indicates that prioritizing some subproblems at specific points of the search can improve the performance of MOEA/D. This issue is commonly addressed using resource allocation (RA) techniques.

Priority functions are used in resource allocation to determine preferences between subproblems. These functions take information about the progress of the search and return priority values that are then used to change the distribution of computational resources among subproblems at any given iteration [13]. They also allow the design of MOEA/D variants that allocate more resources on any desired solution characteristics [10], such as diversity or robustness [14].

Priority functions mediate the distribution of computational resources using a thresholding operation. At any given iteration t, let u_i^t indicate the priority function value attributed to the i-th subproblem, and v^t be a threshold value. The subset of solutions selected for a variation on that iteration is defined as the subproblems for which $u_i^t \geq v^t$.

3 The Importance of Population Size

The population size is one of the essential parameters of Evolutionary Algorithms. It influences the dynamics of those algorithms during the execution [15]. While the right choice can lead to considerable improvements in the performance of an algorithm, a wrong choice of population size might have the contrary effect [9,15]. For example, choosing a population too small can cause premature convergence, and one of the reasons is that a small population size might prevent the localization of optimal solutions. However, a bigger population size may cause the algorithm to waste computational resources, a critical issue in computationally costly problems [16].

Moreover, the proper choice of population size depends on the characteristics of different problems: their difficulty, the presence of many local-optima and the shape of the Pareto Front [9, 15, 17].

In MOEA/D, the choice of the population size also defines the number of sub-problems. Furthermore, each one of these sub-problems might have different characteristics at a given MOP.

We reason that resource allocation can mitigate the burden of choosing the right population size in MOEA/D. That is because MOEA/D with resource allocation techniques benefits from maintaining a big population size and updating very few solutions at each iteration [9, 12]. That is, MOEA/D can maintain and improve a substantially large population at a minimal cost, the cost of updating those few selected solutions, because MOEA/D with RA techniques only updates this few solutions.

Consequently, in this work, we aim to answer the question of how much of the performance improvements observed in MOEA/D with Resource Allocation is due to update only a small number of solutions from a larger population.

4 The n-Partial Update Strategy

To verify whether there is a positive effect in limiting the number of solutions updated at each iteration, and to investigate the extent of this effect, we use the n-partial update strategy (PS). This strategy selects n solutions to update at each iteration. Notice that the PS strategy always includes the solutions of the boundary weight vectors, one for each objective.

The reason for always selecting the boundaries is because they have an impact on the coordinates of the reference point z^* used by the scalarizing function [18]. Algorithm 1 details the pseudocode of the MOEA/D-DE [5], using the n-partial update strategy (MOEA/D-PS).

Notice that the standard MOEA/D, as well as variants such as MOEA/D-DE can be in-

Algorithm 1 MOEA/D-PS (MOEA/D-DE with n Partial Update Strategy)

Input: n, Termination criteria, MOEA/D-DE parameters.
Initialize MOEA/D-DE variables (e.g. weight vectors, set of solutions, etc.)
while *Termination criteria* **do**
    $u_i \leftarrow$ rand() ▷ Vector of random values
    Sample n subproblems given u_i
    for i = 1 to N **do** ▷ Number of subproblems
        if subproblem i was sampled **then**
            Generate new candidate y for subproblem i.
            Evaluate the new candidate solution.
        end if
    end for
    Update the set of solutions.
end while

stantiated from Algorithm 1 by setting n = total number of subproblems. The only difference that the n-partial update strategy introduces in the base algorithm is that only a few subproblems are updated at any given iteration, regulated by the value of n.

It is relevant to observe that subproblems that are not selected by the n-partial update strategy at a given iteration may still have their incumbent solutions updated. Resource allocation in MOEA/D-PS, MOEA/D-DE with the n-partial update strategy, affects only the variation step, not the replacement one; thus, subproblems not selected for variation may receive new candidate solutions, e.g., generated for a neighbouring subproblem.

5 Comparison Study

To demonstrate the relationship between resource allocation strategies and population size, we perform the following experimental study. We compare the MOEA/D-PS against two configurations of MOEA/D with no resource allocation: the first has a big population, the same size as the population size of MOEA/D-PS, and the second has a small population, the same size as the number of solutions updated by MOEA/D-PS at each iteration. This experiment will show that MOEA/D-PS has characteristics of both MOEA/D configurations.

5.1 Benchmark Problems

We use the inverted scalable DTLZ benchmark (DTLZ^{-1}) set [19], with 2 objectives and with dimension $D = 100$.

- DTLZ1^{-1}: Linear Pareto Front;
- DTLZ2^{-1}: Convex Pareto Front;
- DTLZ3^{-1}: Convex Pareto Front;
- DTLZ4^{-1}: Convex Pareto Front.

5.2 Experimental Parameters

We used the MOEA/D-DE parameters as they were introduced in the work of Li and Zhang [2] in all tests. Table 1 summarizes the experimental parameters.

Table 1: Experimental parameter settings.

MOEA/D-DE parameters	Value
DE mutation parameter	$F = 0.25$
Polynomial mutation parameters	$\eta_m = 20$ $p_m = 0.01$
Restricted Update parameter	$nr = 2$
Locality parameter	$\delta_p = 0.9$
Neighborhood size	$T = 20$

SLD decomposition parameter	Value
MOEA/D-PS	$h = 449$
MOEA/D with big pop.	$h = 449$
MOEA/D with small pop.	$h = 49$

Population size	Value
MOEA/D-PS	500
MOEA/D with big pop.	500
MOEA/D with small pop.	50

Resource allocation parameter	Value
ps	50, 10% of the pop. size with the boundary weight vectors

Experiment Parameters	Value
Repeated runs	10
Computational budget	30000 evals.

Details of the parameters can be found in the documentation of package MOEADr and at the

original MOEA/D-DE reference [5, 20, 21]. All objectives were linearly scaled at every iteration to the interval $[0, 1]$, and the Weighted Tchebycheff scalarization [22] function was used.

5.3 Experimental Evaluation

We compare the results of the different strategies using the Hypervolume (HV, higher is better) indicator. For the calculation of HV, the objective function was scaled to the $(0, 1)$ interval, with reference points set to $(1, 1)$.

6 Results

Figure 1 depicts the final approximated Pareto Front of MOEA/D with small population and with big population as well as MOEA/D-PS, in the DTLZ1^{-1} and DTLZ3^{-1}. As we can see, MOEA/D-PS provides a good trade-off between the other variants with different population sizes, since this MOEA/D variant can find a well spread approximation for the Pareto Front with good convergence, independently of the MOPs in question.

Table 2 shows the mean results obtained by the MOEA/D-PS with $ps = 50$ and MOEA/D without resource allocation with (1) small population size and (3) big population size, for all test problems. It is clear that the results of MOEA/D-PS are more stable when compared to the other methods, since MOEA/D-PS achives overall good results in most MOPs, in terms of HV and on the proportion of non-dominated solutions (NDOM) that it returns in the final population. Looking at the proportion of non-dominated solutions (NDOM) in Table 2, we see that updating a subset of solutions from a larger population at each iteration resulted in the highest value on all functions.

6.1 Empirical Attainment Performance

The empirical attainment function (EAF) allows the examination of solution many sets of different runs of an algorithm and it can illustrate where and by how much the outcomes of two algorithms differ in the objective space [23]. The EAF is based on attainment surface. The attained surface separates the objective space in two regions, one where the objective space is dominated (attained) by solutions of many sets and another, where the objective space that is not dominated by those same solutions [24, 25]. For example, the median attainment surface shows regions where dominated by at least half of the runs [23].

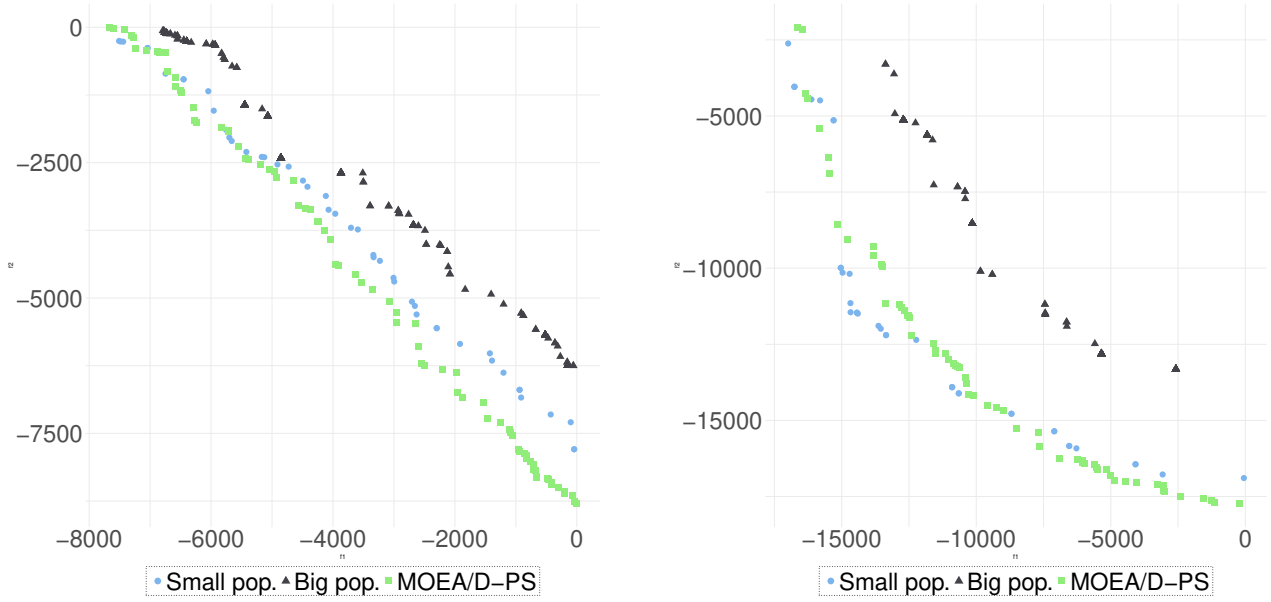
Table 2: Means and standard errors for HV and proportion of non-dominated solutions (NDOM), for each algorithm-problem pair. The best point estimate for each problem is highlighted.

HV			
	MOEA/D-PS	Big Pop.	Small pop.
DTLZ1^{-1}	**0.63 (0.01)**	0.51 (0.01)	0.63 (0.02)
DTLZ2^{-1}	**0.8 (0)**	**0.8 (0)**	0.79 (0)
DTLZ3^{-1}	0.73 (0.04)	0.49 (0.01)	**0.75 (0.03)**
DTLZ4^{-1}	**0.82 (0)**	**0.82 (0)**	0.81 (0)

NDOM			
	MOEA/D-PS	Big Pop.	Small pop.
DTLZ1^{-1}	**499.3 (2.21)**	248.9 (46.59)	48.2 (1.87)
DTLZ2^{-1}	**500 (0)**	488.6 (18.03)	50 (0)
DTLZ3^{-1}	**500 (0)**	178.6 (74.15)	37.3 (7.12)
DTLZ4^{-1}	**500 (0)**	484.5 (17.82)	50 (0)

Examining the differences between EAFs. Figure 2 depicts the differences between the EAFs of MOEA/D-PS and the EAF the MOEA/D with different population sizes, on DTLZ1^{-1}. The lower line, in all Figures, shows the the global best set of solutions attained over all runs of all algorithms (grand best attainment surface) while the upper line shows solutions that are always dominated (grand worst attainment surface).

Figure 2 shows in shades of red, for the DTLZ1^{-1}. Looking at the top part of the Figure (a - left side), we can see that the region where the EAF of MOEA/D with small population performs better than MOEA/D-PS, attaining this region at least 20% of the runs. On the right side of panel (a - right side) we can see that the differences are in the opposite direction. These EAF differences favors MOEA/D-PS, that attains a region that has solutions in over than 40% of the runs. Now looking at the bottom of the Figure (b - left side), we can see that MOEA/D with big population cannot find a EAF region that is better than MOEA/D-PS. In agreement, MOEA/D-PS attains all the dif-



(a) DTLZ1^{-1}.

(b) DTLZ3^{-1}.

Fig. 1: Approximated Pareto Fronts (PF) of the three different variants of MOEA/D. The PF of the MOEA/D-PS is shown as the green squares, the PF of MOEA/D with a small population is shown as the blue circles, and the PF of the MOEA/D with a big population is shown as the grey triangles. It is clear that MOEA/D-PS can find a well spread PF with good convergence speed.

ference between the EAFs.

6.2 Anytime Performance

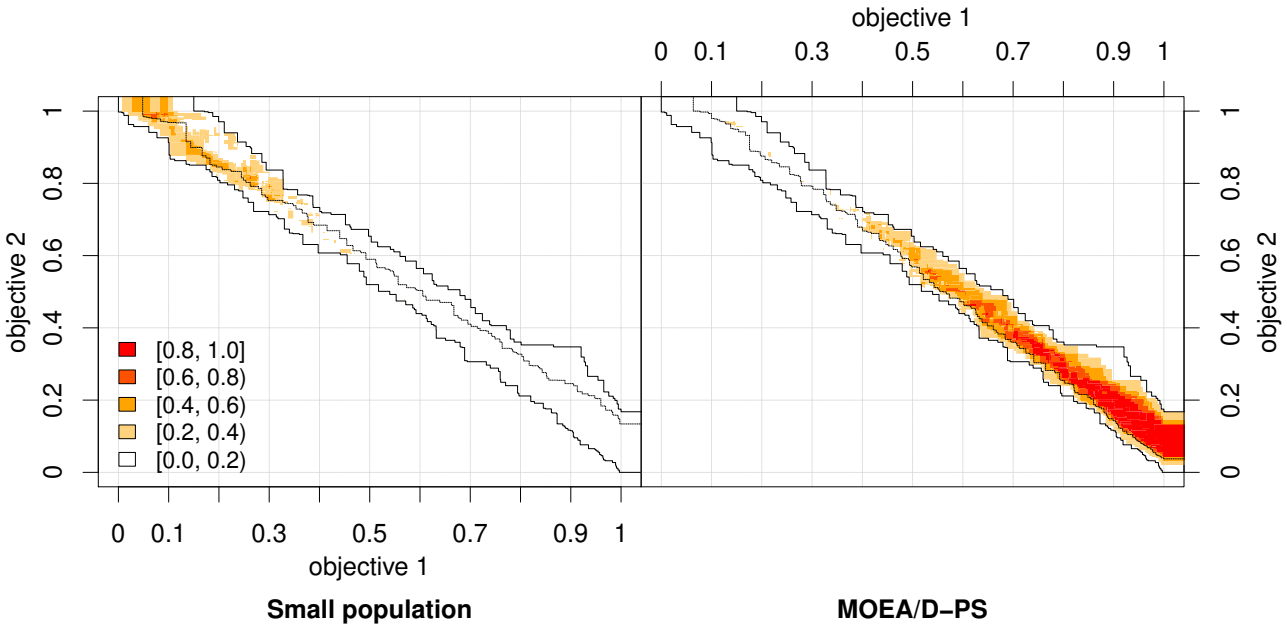
Besides providing good final results, it is often desired that an MOEA be capable of returning a set of reasonably good solutions if interrupted at any time during the search [26, 27]. We analyzed the anytime performance effects in terms of HV values to investigate the impact of different population sizes and MOEA/D-PS.

Figures 3 illustrates the anytime performance of the three different MOEA/D variants hypervolume (higher is better) for all problems. These results indicate that, in shorter time frames, MOEA/D-PS shows an improvement over MOEA/D with a big population because MOEA/D-PS has better convergence speed, similar to the speed of the MOEA/D with a small population, without shortcomings in terms of quality of the Pareto Front. On the other hand, for certain problems, if the optimization is performed with a large number of functions evaluations, MOEA/D with a big population size might eventually find better results than MOEA/D-PS.

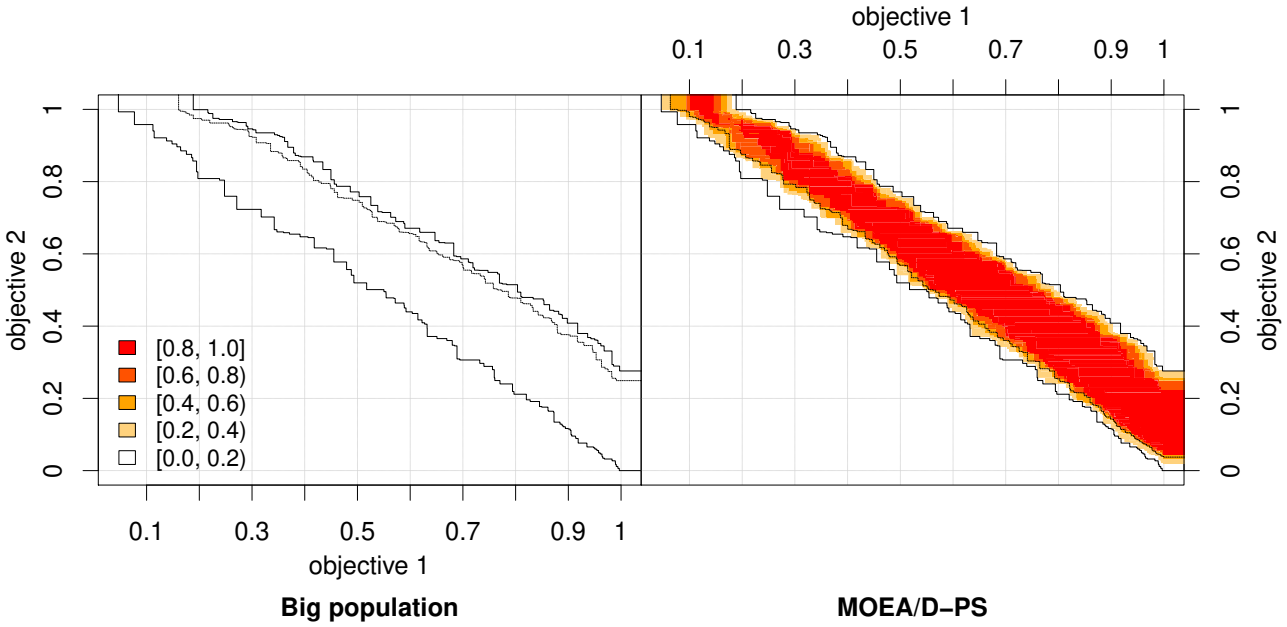
7 Conclusion

In this work, we compare the effect of Resource Allocation strategies in MOEA/D against with different population size settings. This is motivated by the realization that the n-partial update strategy uses its control parameter (ps) to regulate the proportion of the population that is selected for variation at any iteration.

We found strong evidence that MOEA/D-PS has a more stable performance independently of the MOP in question since MOEA/D-PS always performs as one of the best algorithms in our experiments. The Empirical Attainment Function (EAF) results illustrate the hybrid behaviour of MOEA/D-PS, as we can observe that it can find better EAF regions in all problems tested when compared to both MOEA/D with small or big population. The Anytime Analysis of the algorithms show that MOEA/D-PS has a similar behavior as MOEA/D with a small population, quickly finding its best results, while maintaining the benefit of a wider exploration of the search space that is characteristic of MOEA/D with a big population.

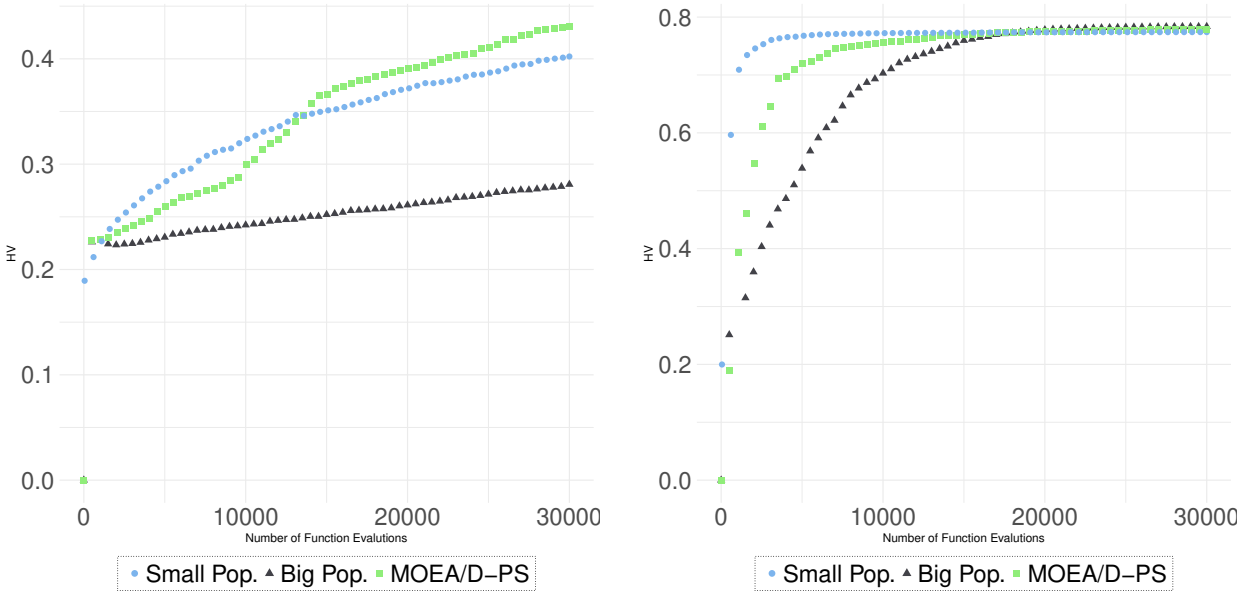


(a) MOEA/D-PS (right) performs better towards solutions for objective 2 (low values on the y-axis). On the other hand, MOEA/D with small population (left) performs better towards solutions for objective 1 (low values on the x-axis).



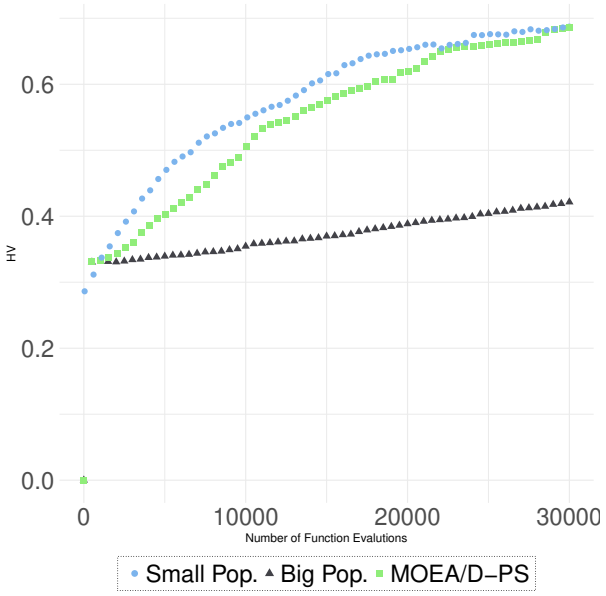
(b) MOEA/D-PS (right) performs better at all regions of the objective space.

Fig. 2: Differences between the EAFs of MOEA/D with small population (top,left) and MOEA/D-PS (top, right); and MOEA/D with big population (bottom,left) and MOEA/D-PS (bottom, right). The red level encodes the magnitude of the observed difference, on the DTLZ1^{-1}.

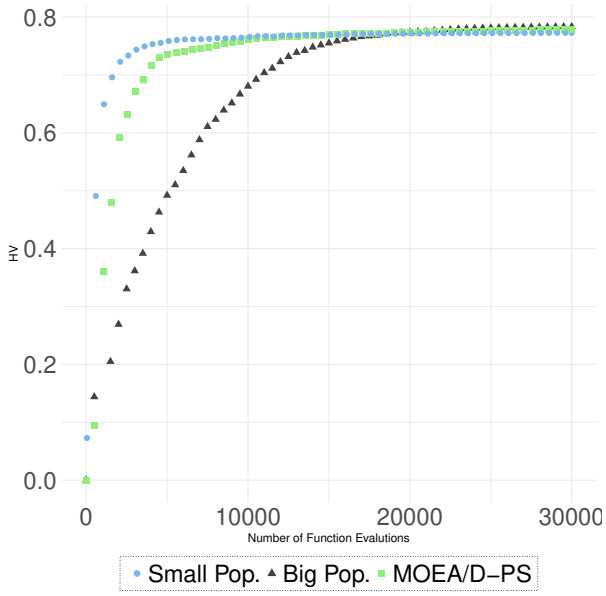


(a) DTLZ1^{-1}.

(b) DTLZ2^{-1}.



(c) DTLZ3^{-1}.



(d) DTLZ4^{-1}.

Fig. 3: Anytime HV (higher is better) performance of MOEA/D with PS is shown as the green squares, MOEA/D with population size 500 is shown as the grey triangles and MOEA/D with population size equals to 50 is shown as the blue circles. The anytime performance of MOEA/D-PS is similar to the anytime performance of MOEA/D with a small population. In DTLZ2^{-1} and DTLZ4^{-1}, MOEA/D-PS performs much similar by MOEA/D with a big population after 15000 evaluations, around 2/3 of the search. This dynamic indicates that MOEA/D-PS has almost the same convergence speed as MOEA/D with a small population and that its performance will eventually be surpassed by MOEA/D with a big population.

MOEA/D-PS progresses with the search as an algorithm with a small population size and MOEA/D-PS can explore the search space as an algorithm with a big population size. The reason relies on the relationship between different populations and their advantages in solving a MOP. That is, a small population size is able to approach the Pareto Front quickly, but might not be able to explore and cover this Pareto Front given its limited size; however, a larger population size will likely be better at this task, at a higher cost [15]. In summary, MOEA/D-PS provides a simple yet efficient approach to mitigate common problems related to the choice of population size, such as the likely waste of computation resources induced by a large population size or the premature stagnation of a small population size [28].

As future works, we highlight two directions to extend, improve and explain MOEA/D-PS. The first is whether MOEA/D-PS would benefit from adapting the *ps* value throughout the search, especially in cases where the performance improvements between iterations are minimal. The second is to study the algorithm performance and behaviour on constrained MOPs, such as the recent CEC'19 problems as the JNPSEC simulation-based MOPs.

参考文献

1) A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh, "A survey of multiobjective evolutionary algorithms based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 440–462, 2017.

2) H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on evolutionary computation*, vol. 13, no. 2, pp. 284–302, 2009.

3) E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2004, pp. 832–842.

4) Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

5) Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC'09 unconstrained mop test instances," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 203–208.

6) A. Zhou and Q. Zhang, "Are all the subproblems equally important? Resource allocation in decomposition-based multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 52–64, 2016.

7) Q. Kang, X. Song, M. Zhou, and L. Li, "A collaborative resource allocation strategy for decomposition-based multiobjective evolutionary algorithms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

8) Y. Lavinas, C. Aranha, and T. Sakurai, "Using diversity as a priority function for resource allocation on MOEA/D," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2019, pp. 215–216.

9) G. Pruvost, B. Derbel, A. Liefooghe, K. Li, and Q. Zhang, "On the combined impact of population size and sub-problem selection in MOEA/D," in *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*. Springer, 2020, pp. 131–147.

10) Y. Lavinas, C. Aranha, and M. Ladeira, "Improving resource allocation in MOEA/D with decision-space diversity metrics," in *Theory and Practice of Natural Computing*. Cham: Springer International Publishing, 2019, pp. 134–146.

11) P. Wang, W. Zhu, H. Liu, B. Liao, L. Cai, X. Wei, S. Ren, and J. Yang, "A new resource allocation strategy based on the relationship between subproblems for MOEA/D," *Information Sciences*, vol. 501, pp. 337–362, 2019.

12) Y. Lavinas, C. Aranha, M. Ladeira, and F. Campelo, "MOEA/D with random partial update strategy," *arXiv preprint arXiv:2001.06980*, 2020.

13) X. Cai, Y. Li, Z. Fan, and Q. Zhang, "An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 508–523, 2015.

14) F. Goulart, S. T. Borges, F. C. Takahashi, and F. Campelo, "Robust multiobjective optimization using regression models and linear subproblems," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2017, pp. 569–576.

15) T. Glasmachers, B. Naujoks, and G. Rudolph, "Start small, grow big? Saving multi-objective function evaluations," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2014, pp. 579–588.

16) T. Kohira, H. Kemmotsu, O. Akira, and T. Tatsukawa, "Proposal of benchmark problem based on real-world car structure design optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2018, pp. 183–184.

17) M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM computing surveys (CSUR)*, vol. 45, no. 3, pp. 1–33, 2013.

18) P. Wang, W. Zhu, H. Liu, B. Liao, L. Cai, X. Wei, S. Ren, and J. Yang, "A new resource allocation strategy based on the relationship between subproblems for MOEA/D," *Information Sciences*, vol. 501, pp. 337–362, 2019.

19) H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, "Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 169–190, 2016.

20) F. Campelo and C. Aranha, "MOEADr: Component-wise MOEA/D implementation," URL https://cran.R-project.org/package=MOEADr, 2018, r package version 1.2.0.

21) F. Campelo, L. Batista, and C. Aranha, "The MOEADr package: A component-based framework for multiobjective evolutionary algorithms based on decomposition," *Journal of Statistical Software*, 2020, in press. Available from: https://arxiv.org/abs/1807.06731.

22) K. Miettinen, "Introduction to multiobjective optimization: Noninteractive approaches," in *Multiobjective optimization*. Springer, 2008, pp. 1–26.

23) M. López-Ibánez, L. Paquete, and T. Stützle, "Exploratory analysis of stochastic local search algorithms in biobjective optimization," in *Experimental methods for the analysis of optimization algorithms*. Springer, 2010, pp. 209–222.

24) C. M. Fonseca and P. J. Fleming, "On the performance assessment and comparison of stochastic multiobjective optimizers," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1996, pp. 584–593.

25) V. G. Da Fonseca, C. M. Fonseca, and A. O. Hall, "Inferential performance assessment of stochastic optimisers and the attainment function," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2001, pp. 213–225.

26) R. Tanabe, H. Ishibuchi, and A. Oyama, "Benchmarking multi-and many-objective evolutionary algorithms under two optimization scenarios," *IEEE Access*, vol. 5, pp. 19 597–19 619, 2017.

27) A. Radulescu, M. López-Ibánez, and T. Stützle, "Automatically improving the anytime behaviour of multiobjective evolutionary algorithms," in *International*

Conference on Evolutionary Multi-Criterion Optimization. Springer, 2013, pp. 825–840.

28) Q. Lin, Q. Zhu, N. Wang, P. Huang, W. Wang, J. Chen, and Z. Ming, "A multi-objective immune algorithm with dynamic population strategy," *Swarm and Evolutionary Computation*, vol. 50, p. 100477, 2019.